Learning from Data For Linguists
Lecture 2: Evaluation and Naive Bayes

Malvina Nissim and Johannes Bjerva
m.nissim@rug.nl, j.bjerva@rug.nl

ESSLLI 2016, 23 August 2016

Evaluation

Evaluation of results

$\rightarrow$ is the system really able to generalise?

Evaluation of results

→ is the system really able to generalise?

- the test set is equipped with **class labels**, manually assigned (gold standard)
- for each instance in the test set, we compare the class predicted by the classifier with the class specified in the gold standard

have to predict: steal or boil?

```
trying,to,upon,the,?
began,to,partners,and,?
them,to,workers,or,?
pate,or,it,in,?
gently,and,spoonfuls,of,?
let,them,for,3,?
```

you have your gold standard:

```
trying,to,upon,the,steal
began,to,partners,and,steal
them,to,workers,or,steal
pate,or,it,in,boil
gently,and,spoonfuls,of,boil
let,them,for,3,boil
```

you compare:

gold	prediction
`trying,to,upon,the,steal`	`trying,to,upon,the,steal`
`began,to,partners,and,steal`	`began,to,partners,and,boil`
`them,to,workers,or,steal`	`them,to,workers,or,boil`
`pate,or,it,in,boil`	`pate,or,it,in,boil`
`gently,and,spoonfuls,of,boil`	`gently,and,spoonfuls,of,steal`
`let,them,for,3,boil`	`let,them,for,3,boil`

- how do we *measure* performance?
- when is a model good enough?

Evaluation measures

- accuracy: percentage of correct decisions overall

Evaluation measures

- accuracy: percentage of correct decisions overall

gold	prediction
`trying,to,upon,the,steal`	`trying,to,upon,the,steal`
`began,to,partners,and,steal`	`began,to,partners,and,boil`
`them,to,workers,or,steal`	`them,to,workers,or,boil`
`pate,or,it,in,boil`	`pate,or,it,in,boil`
`gently,and,spoonfuls,of,boil`	`gently,and,spoonfuls,of,steal`
`let,them,for,3,boil`	`let,them,for,3,boil`

accuracy = ?

Evaluation measures

- accuracy: percentage of correct decisions overall

gold	prediction
`trying,to,upon,the,steal`	`trying,to,upon,the,steal`
`began,to,partners,and,steal`	`began,to,partners,and,boil`
`them,to,workers,or,steal`	`them,to,workers,or,boil`
`pate,or,it,in,boil`	`pate,or,it,in,boil`
`gently,and,spoonfuls,of,boil`	`gently,and,spoonfuls,of,steal`
`let,them,for,3,boil`	`let,them,for,3,boil`

accuracy $= 3/6 = 50\%$

Evaluation measures

Consider class "X"

- true positive (**TP**): X classified as X
- true negative (**TN**): ¬X classified as ¬X
- false positive (**FP**): ¬X classified as X
- false negative (**FN**): X classified as ¬X

Evaluation measures

Consider class "X"

- true positive (**TP**): X classified as X
- true negative (**TN**): ¬X classified as ¬X
- false positive (**FP**): ¬X classified as X
- false negative (**FN**): X classified as ¬X

Consider class "steal"

- true positive (**TP**): steal classified as steal
- true negative (**TN**): boil classified as boil
- false positive (**FP**): boil classified as steal
- false negative (**FN**): steal classified as boil

confusion matrix

<u>steal</u>

		prediction	
		steal	boil
gold standard	steal	**TP**	
	boil		**TN**

confusion matrix

steal

prediction

gold standard		steal	boil
	steal	**TP**	**FN**
	boil	**FP**	**TN**

Evaluation measures

- **precision**$_X$:
  correct decisions over instances assigned to class "X"
  $TP/(TP+FP)$

- **recall**$_X$:
  correct assignments to class "X" over all instances of class "X" in test set
  $TP/(TP+FN)$

- **f-score**$_X$:
  combined measure of precision and recall
  $F = \frac{2PR}{P+R}$

Evaluation measures

gold	prediction
`trying,to,upon,the,steal`	`trying,to,upon,the,steal`
`began,to,partners,and,steal`	`began,to,partners,and,boil`
`them,to,workers,or,steal`	`them,to,workers,or,boil`
`pate,or,it,in,boil`	`pate,or,it,in,boil`
`gently,and,spoonfuls,of,boil`	`gently,and,spoonfuls,of,steal`
`let,them,for,3,boil`	`let,them,for,3,boil`

$\text{precision}_{steal} = ?$
$\text{recall}_{steal} = ?$

Evaluation measures

gold	prediction
trying,to,upon,the,steal	trying,to,upon,the,steal
began,to,partners,and,steal	began,to,partners,and,boil
them,to,workers,or,steal	them,to,workers,or,boil
pate,or,it,in,boil	pate,or,it,in,boil
gently,and,spoonfuls,of,boil	gently,and,spoonfuls,of,steal
let,them,for,3,boil	let,them,for,3,boil

$\text{precision}_{steal} = 1/2 = 50\%$
$\text{recall}_{steal} =$

Evaluation measures

gold	prediction
trying,to,upon,the,steal	trying,to,upon,the,steal
began,to,partners,and,steal	began,to,partners,and,boil
them,to,workers,or,steal	them,to,workers,or,boil
pate,or,it,in,boil	pate,or,it,in,boil
gently,and,spoonfuls,of,boil	gently,and,spoonfuls,of,steal
let,them,for,3,boil	let,them,for,3,boil

$\text{precision}_{steal} = 1/2 = 50\%$
$\text{recall}_{steal} = 1/3 = 33\%$

Evaluation measures

what is good enough?

- upperbound: inter-annotator agreement
- baseline: performance of basic, simple model
  for example: **assignment of most frequent class in data set**
  - sense_1 $9/10$ and sense_2 $1/10$
  - sense_1 $6/10$ and sense_2 $4/10$

What happens in learning, then?

- the learning algorithm observes **given examples**
- it tries to find common patterns that explain the data: it tries to **generalise** so that predictions can be made for **new examples**
- exactly how this is done depends on what **algorithm we are using**

keywords here:

- given/new examples
  - the settings of a learning experiment are important
- generalising
  - what does it mean to generalise well?
- algorithm we are using
  - **we are going to see one now**

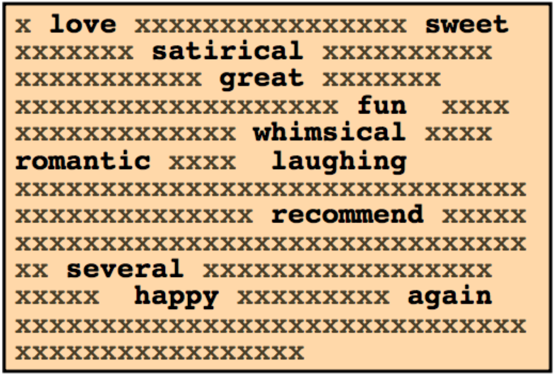
Naive Bayes

Naive Bayes classification

- simple classification method based on Bayes rule
- relies on a simple representation of documents: bag of words

Naive Bayes classification

- simple classification method based on Bayes rule
- relies on a simple representation of documents: bag of words



Naive Bayes classification

- simple classification method based on Bayes rule
- relies on a simple representation of documents: bag of words

great	2
love	2
recommend	1
laugh	1
happy	1
...	...

Conditional probability

- conditional probability of an event: a probability obtained with the **additional information** that **some other event** has already occurred
- new information is used to revise the probability of the initial event
- **prior** vs **posterior probability**
    - prior: probability obtained "as things stand", before any additional information is acquired
    - posterior: probability value which has been revised by using additional information

Example

in a corpus: happy tweets = 45% ; sad tweets = 55%

- I select one instance, how probable is it to be happy?
- The tweet contains the word "cheerful".
  "cheerful" occurs in 70% of happy tweets,
  and in 25% of sad tweets.
  Does the probability now change?
- which one is prior and which one posterior?

Example

in a corpus: happy tweets = 45% ; sad tweets = 55%

- I select one instance, how probable is it to be happy?
- The tweet contains the word "cheerful".
  "cheerful" occurs in 70% of happy tweets,
  and in 25% of sad tweets.
  Does the probability now change?
- which one is prior and which one posterior?

Example

in a corpus: happy tweets = 45% ; sad tweets = 55%

- I select one instance, how probable is it to be happy?
- The tweet contains the word "cheerful".
  "cheerful" occurs in 70% of happy tweets,
  and in 25% of sad tweets.
  Does the probability now change?
- which one is prior and which one posterior?

Bayes' Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: a given class (happy)
- i: a given instance ("I always feel cheerful on Friday evening")

Bayes' Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: a given class (happy)
- i: a given instance ("I always feel cheerful on Friday evening")

- $p(c_j|i)$ = pr of instance i being in class c_j
  how likely is it this given tweet from the corpus is happy?

Bayes' Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: a given class (happy)
- i: a given instance ("I always feel cheerful on Friday evening")

- $p(c_j|i)$ = pr of instance i being in class c_j
  how likely is it this given tweet from the corpus is happy?
- $p(i|c_j)$ = (*likelihood function*) = pr of generating instance i given class c_j (happy)
  given class c_j (happy) how likely is it to get i?
  TRUE POSITIVE: 70%

Bayes' Theorem

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j){\color{red}p(i|\neg c_j)}]}$$

- c_j: a given class (happy)
- i: a given instance ("I always feel cheerful on Friday evening")

- $p(c_j|i)$ = pr of instance i being in class c_j
  how likely is it this given tweet from the corpus is happy?
- $p(i|c_j)$ = (*likelihood function*) = pr of generating instance i given class c_j (happy)
  given class c_j (happy) how likely is it to get i?
  TRUE POSITIVE: 70%
- ${\color{red}p(i|\neg c_j)}$ = pr of generating instance i given $\neg c_j$ (sad)
  given $\neg c_j$ (sad) how likely is it to get i?
  FALSE POSITIVE: 25%

Worked Example

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets

Worked Example

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets

Worked Example

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{[p(c_j)p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening"
  (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(c_j) =$

Worked Example

$$p(c_j|i) = \frac{p(i|c_j) \cdot 0.45}{[0.45 \cdot p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(c_j) = 0.45$

Worked Example

$$p(c_j|i) = \frac{p(i|c_j) \cdot 0.45}{[0.45 \cdot p(i|c_j)] + [p(\neg c_j)p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(c_j) = 0.45$
- $p(\neg c_j) =$

Worked Example

$$p(c_j|i) = \frac{p(i|c_j) \cdot 0.45}{[0.45 \cdot p(i|c_j)] + [0.55 \cdot p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening"
  (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(c_j) = 0.45$
- $p(\neg c_j) = 0.55$

Worked Example

$$p(c_j|i) = \frac{p(i|c_j) \cdot 0.45}{[0.45 \cdot p(i|c_j)] + [0.55 \cdot p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(i|c_j)$ = given class c_j (happy) how likely is it to get i?

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(i|c_j)$ = given class c_j (happy) how likely is it to get i? 70%

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot p(i|\neg c_j)]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy $=$ 45%
- said $=$ 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(i|c_j)$ $=$ given class c_j (happy) how likely is it to get i? 70% TRUE POSITIVE
- $p(i|\neg c_j)$ $=$ given class $\neg c_j$ (sad) how likely is it to get i?

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot \color{red}{0.25}]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(i|c_j)$ = given class c_j (happy) how likely is it to get i? 70% TRUE POSITIVE
- $\color{red}{p(i|\neg c_j)}$ = given class $\neg c_j$ (sad) how likely is it to get i? $\color{red}{25\%}$

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot 0.25]}$$

- c_j: happy
- i: "I always feel cheerful on Friday evening" (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(i|c_j)$ = given class c_j (happy) how likely is it to get i? 70% TRUE POSITIVE
- $p(i|\neg c_j)$ = given class $\neg c_j$ (sad) how likely is it to get i? 25% FALSE POSITIVE

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot 0.25]} = 0.70$$

- c_j: happy
- i: "I always feel cheerful on Friday evening"
  (represented by feature value "cheerful")
- happy = 45%
- said = 55%
- "cheerful" in 70% of happy tweets
- "cheerful" in 25% of sad tweets
- $p(i|c_j)$ = given class c_j (happy) how likely is it to get i? 70%
  TRUE POSITIVE
- $p(i|\neg c_j)$ = given class $\neg c_j$ (sad) how likely is it to get i? 25%
  FALSE POSITIVE

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot 0.25]} = 0.70$$

- **prior** probability of i as *happy* $= 0.45$
- **posterior** probability of i as *happy* $= 0.70$

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot 0.25]} = 0.70$$

- **prior** probability of i as *happy* $= 0.45$
- **posterior** probability of i as *happy* $= 0.70$

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

$p(i)$ = the probability of i ("cheerful") overall

Worked Example

$$p(c_j|i) = \frac{0.70 \cdot 0.45}{[0.45 \cdot 0.70] + [0.55 \cdot 0.25]} = 0.70$$

- **prior** probability of i as *happy* $= 0.45$
- **posterior** probability of i as *happy* $= 0.70$

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

$p(i)$ = the probability of i ("cheerful") overall

$70\% \cdot 45\% + 25\% \cdot 55\%$

Worked Example

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

how do we make a classifier out of this?
we need to pick a class c out of a set of possible class values.

Worked Example

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

how do we make a classifier out of this?
we need to pick a class c out of a set of possible class values.

we add a *decision rule*: maximum a posteriori (*map*)

$$c_{map} = \arg\max_{c \in C} \frac{p(i|c) \cdot p(c)}{p(i)}$$

Worked Example

$$p(c_j|i) = \frac{p(i|c_j)p(c_j)}{p(i)}$$

how do we make a classifier out of this?
we need to pick a class c out of a set of possible class values.

we add a *decision rule*: maximum a posteriori (*map*)

$$c_{map} = \underset{c \in C}{\arg\max} \frac{p(i|c) \cdot p(c)}{p(i)}$$

note that because we only need to *compare* values, we can drop the denominator, which basically serves as normalising function

$$c_{map} = \underset{c \in C}{\arg\max}\, p(i|c) \cdot p(c)$$

Two issues

- zeros and smoothing (Laplace or add-one smoothing)
- underflow

Two issues

- zeros and smoothing (Laplace or add-one smoothing)
  it can happen that some values are zero. To prevent this problem in the calculations, the value 1 is added to all observed counts
- underflow

Two issues

- zeros and smoothing (Laplace or add-one smoothing)
- underflow
  posterior probabilities are usually very very small, especially with lots of features (think of a bag-of-words approach in text classification).
  This is called the *underflow* problem
  For this reason, most implementations of a NB classifier use the log of the probabilities

let's get things into practice

Evaluation poll

		Prediction	
		Cat	Dog
Actual	Cat	15	35
	Dog	40	10

http://etc.ch/aJYc

Evaluation poll -results

http://directpoll.com/r?XDbzPBd3ixYqg8V6YIo1K6SeELEgWl9oEnTt4iBkI

Getting the updated code

- Approach 1: Use *git* (updateable, recommended if you have *git*)
  1. In your terminal, type: 'git clone https://github.com/bjerva/esslli-learning-from-data-students.git'
  2. Followed by 'cd esslli-learning-from-data-students'
  3. Whenever the code is updated, type: 'git pull'
- Approach 2: Download a zip (static)
  1. Download the zip archive from: `https://github.com/bjerva/esslli-learning-from-data-students/archive/master.zip`
  2. Whenever the code is updated, download the archive again.

Feature poll

http://etc.ch/Mghf

Feature poll - results

http://directpoll.com/r?XDbzPBd3ixYqg8rnZ5Rw6iaSm92UKZxc2bHhsWzY6

Running an experiment

1. Navigate to your 'esslli-learning-from-data-students' (using cd in the terminal)
2. To extract features:
   python feature_extractor.py −−csv data/trainset-sentiment.csv −−fname sentiment −−nwords 1

Running an experiment

1. Navigate to your 'esslli-learning-from-data-students' (using cd in the terminal)
2. To extract features:
   python feature_extractor.py −−csv data/trainset-sentiment.csv −−fname sentiment −−nwords 1

   → This is a bag-of-word model!

Running an experiment

1. Navigate to your 'esslli-learning-from-data-students' (using cd in the terminal)
2. To extract features:
   python feature_extractor.py --csv data/trainset-sentiment.csv --fname sentiment --nwords 1

   → This is a bag-of-word model!
3. To learn using these features:
   python learn_from_data.py --npz sentiment.npz --algorithms nb

Adding features

1. Navigate to your 'esslli-learning-from-data-students' (using cd in the terminal)
2. To extract features:
   python feature_extractor.py −−csv data/trainset-sentiment.csv −−fname sentiment −−nwords 1
3. To learn using these features:
   python learn_from_data.py −−npz sentiment.npz −−algorithms nb

Using more features

- −−nwords n, extract word ngrams of order n (e.g. −−nwords 2)
- −−nchars n, extract character ngrams of order n (e.g. −−nchars 3)
- −−features x y z, extract features with the given names (e.g. −−features gender-cat time-cat)

End poll

http://directpoll.com/r?XDbzPBd3ixYqg81LygsVoSIvClR6cnLre6kxM2M3